AMENDMENTS TO THE CLAIMS

Claim 1 (currently amended): In a processing unit including nonvolatile memory for storing executable code, the nonvolatile memory comprising a vital region for storing vital code and a region for storing less vital code, an apparatus for executing an update program for managing updates to the vital region of the nonvolatile memory with update code, the apparatus comprising:

a processor programmed for

issuing an identifier command to the nonvolatile memory;

receiving from the nonvolatile memory an identifier that identifies ~~identifying~~ a memory type of the nonvolatile memory to determine appropriate commands to be sent;

performing a write test by writing test code into multiple blocks in the less vital region and verifying test status and data of each block to simulate an actual update to the vital region, and

performing an actual write of the update code into the vital region if the write test is successful.

Claim 2 (previously presented): The apparatus as recited in claim 1, each region of the nonvolatile memory of the processing unit having a version number and likewise each region of the update code having a version number, the processor further programmed for:

comparing the version number of the update code and the version number of the nonvolatile memory for a given region, and

executing the update program only if the version number of the update code is greater than the nonvolatile memory code version number.

Claim 3 (original): The apparatus as recited in claim 2, wherein a location within each vital region is allocated to store the version number, the processor further programmed to access this location to retrieve the version number from the nonvolatile memory.

Claim 4 (original): The apparatus as recited in claim 3, the processor further programmed for writing the version number of the update code into the allocated area in that region of nonvolatile memory if the write test is successful.

Claim 5 (original): The apparatus as recited in claim 2, the processor further programmed for terminating the update without performing any write operations if the version number of the update code is less than or equal to the nonvolatile memory version number for each vital region.

Claim 6 (original): The apparatus as recited in claim 2, the processor further programmed for querying a user whether the update program should proceed if the version number of the update code is less than or equal to the nonvolatile memory version number.

Claim 7 (original): The apparatus as recited in claim 1, wherein the vital region is a boot region and the non-vital region is a test region.

Claim 8 (original): The apparatus as recited in claim 1, wherein the vital region is a boot region and the less vital region is an application region.

Claim 9 (original): The apparatus as recited in claim 8, wherein the test code is equivalent to application code currently stored in the application region of nonvolatile memory.

Claim 10 (original): The apparatus as recited in claim 1, wherein the test code is equivalent to the update code.

Claim 11 (original): A host bus adapter (HBA) comprising the apparatus of claim 1, the HBA for implementing upper layer protocols (ULPs).

Claim 12 (original): The HBA of claim 11, further comprising an Internet Small Computer System Interface (iSCSI) or a fibre channel controller circuit.

Claim 13 (original): A host computer comprising the HBA of claim 12.

Claim 14 (original): A storage area network (SAN) comprising the host computer of claim 13, wherein an iSCSI or a fibre channel network is coupled to the iSCSI or fibre channel controller circuit and one or more storage devices are coupled to the iSCSI or fibre channel network.

Claim 15 (currently amended): In a processing unit including nonvolatile memory for storing executable code, the nonvolatile memory comprising a vital region for storing vital code and a region for storing less vital code, a computer program for managing updates to the vital region of the nonvolatile memory with update code, the computer program being stored on a machine readable medium and executable to perform acts comprising:

issuing an identifier command to the nonvolatile memory;

receiving from the nonvolatile memory an identifier that identifies ~~identifying~~ a memory type of the nonvolatile memory to determine appropriate commands to be sent;

performing a write test by writing test code into multiple blocks in the less vital region in nonvolatile memory and verifying test status and data of each block to simulate an actual update to the vital region; and

performing an actual write of the update code into the vital region in nonvolatile memory, if the write test is successful.

Claim 16 (original): The computer program as recited in claim 15, each region of the nonvolatile memory of the processing unit having a version number and the update code having a version number, the computer program further executable to perform acts comprising:

comparing the version number of the update code and the version number of the nonvolatile memory for a given region, and

updating the vital region of the nonvolatile memory only if the version number of the update code is greater than the nonvolatile memory version number.

Claim 17 (original): The computer program as recited in claim 16, wherein a location within each vital region is allocated to store the version number, the computer program further executable to perform acts comprising accessing this location to retrieve the version number from both the nonvolatile memory and the update code.

Claim 18 (original): The computer program as recited in claim 17, the computer program further executable to perform acts comprising writing the version number of the update code into the allocated area in that region of nonvolatile memory if the write test is successful.

Claim 19 (original): The computer program as recited in claim 16, the computer program further executable to perform acts comprising terminating the update program without performing any write operations if the version number of the update code is less than or equal to the nonvolatile memory version number for each vital region.

Claim 20 (original): The computer program as recited in claim 16, the computer program further executable to perform acts comprising querying a user whether the update program should proceed if the version number of the update code is less than or equal to the nonvolatile memory code version number.

Claim 21 (original): The computer program as recited in claim 15, wherein the vital region is a boot region and the non-vital region is a test region.

Claim 22 (original): The computer program as recited in claim 15, wherein the vital region is a boot region and the less vital region is an application region.

Claim 23 (original): The computer program as recited in claim 22, wherein the test code is equivalent to application code currently stored in the application region of nonvolatile memory.

Claim 24 (original): The computer program as recited in claim 15, wherein the test code is equivalent to the update code.

Claim 25 (original): A host bus adapter (HBA) comprising the computer program of claim 15, the HBA for implementing upper layer protocols (ULPs).

Claim 26 (original): The HBA of claim 25, further comprising an Internet Small Computer System Interface (iSCSI) or a fibre channel controller circuit.

Claim 27 (original): A host computer comprising the HBA of claim 26.

Claim 28 (original): A storage area network (SAN) comprising the host computer of claim 27, wherein an iSCSI or fibre channel network is coupled to the iSCSI or fiber channel controller circuit and one or more storage devices are coupled to the iSCSI or fibre channel network.

Claim 29 (currently amended): In a processing unit including nonvolatile memory for storing executable code, the nonvolatile memory comprising a vital region for storing vital code and a region for storing less vital code, a method for executing an update program for managing updates to the vital region of the nonvolatile memory with update code, the method comprising:

issuing an identifier command to the nonvolatile memory;

receiving from the nonvolatile memory an identifier that identifies ~~identifying~~ a memory type of the nonvolatile memory to determine appropriate commands to be sent;

performing a write test by writing test code into multiple blocks in the less vital region and verifying test status and data of each block to simulate an actual update to the vital region, and

performing an actual write of the update code into the vital region if the write test is successful.

Claim 30 (original): The method as recited in claim 29, each region of the nonvolatile memory of the processing unit having a version number and the update code having a version number, the method further comprising:

comparing the version number of the update code and the version number of the nonvolatile memory for a given region, and

executing the update program only if the version number of the update code is greater than the nonvolatile memory code version number.

Claim 31 (original): The method as recited in claim 30, wherein a location within each vital region is allocated to store the version number, the method further comprising accessing this location to retrieve the version number from the nonvolatile memory.

Claim 32 (original): The method as recited in claim 31, further comprising writing the update code version number of the update code into the allocated area in that region of nonvolatile memory if the write test is successful.

Claim 33 (original): The method as recited in claim 30, further comprising terminating the update without performing any write operations if the version number of the update code is less than or equal to the nonvolatile memory code version number for each vital region.

Claim 34 (original): The method as recited in claim 30, further comprising querying a user whether the update program should proceed if the version number of the update code is less than or equal to the nonvolatile memory code version number.

Claim 35 (original): The method as recited in claim 29, wherein the vital region is a boot region and the less vital region is a test region.

Claim 36 (original): The method as recited in claim 29, wherein the vital region is a boot region and the less vital region is an application region.

Claim 37 (original): The method as recited in claim 36, wherein the test code is equivalent to application code currently stored in the application region.

Claim 38 (original): The method as recited in claim 29, wherein the test code is equivalent to the update code.